Comparison of three common software-defined network controllers

Rikie Kartadie[1], **Edy Prayitno**[2]
[1]Department of Computer Engineering, Universitas Teknologi Digital Indonesia, Yogyakarta, Indonesia
[2]Department of Information System, Universitas Teknologi Digital Indonesia, Yogyakarta, Indonesia

Article Info

Article history:

Received Dec 17, 2021
Revised Mar 16, 2022
Accepted Apr 15, 2022

Keywords:

Controller
Floodlight
Open daylight
Open network operating system
Software-defined network

ABSTRACT

The software-defined network (SDN) controller adds and removes the contents of the flow table through secure channels to determine how packets are processed and how the flow table is managed. The controller pays attention to network intelligence and becomes the middle part, where the network manages the transfer data of the aircraft delivered via the OpenFlow (OF) switch. To this end, the controller provides an interface for managing, controlling, and managing this switch flow table. Run tests to calculate controller throughput and latency levels and test using the cbance tool, which can test transmission control protocol (TCP) and user datagram protocol (UDP) protocols. The tests are run by forcing the controller to run at maximum without any additional settings (default settings) in order to use the correct information about the controller's capabilities. Because of this need, you need to test the performance of your controller. In this study, the tests were run on three popular controllers. Test results show that flowed controllers are more stable than open network operating dystem (ONOS) and open daylight (ODL) controllers in managing switch and host loads.

Corresponding Author:

Rikie Kartadie
Department of Computer Engineering, Universitas Teknologi Digital Indonesia
Jl. Raya Janti, Jl. Majapahit No. 143, Jaranan, Banguntapan, Bantul, Yogyakarta 55918, Indonesia
Email: rikie@utdi.ac.id

1. INTRODUCTION

The controller that directly controls the data from the device is a major component of the software-defined network (SDN). The controller is responsible for determining the way to handle packages and manage flow tables by adding and removing contents of flow tables through secure channels. The controller concentrates network intelligence, while the network maintains data forwarding that is distributed through the OpenFlow (OF) switch. For this reason, the controller provides an interface for managing, controlling, and administering flow tables on the switch [1].

The controller performance is one of the tens controllers selection criteria [2], the SDN controller must be able to make flow tables beforehand to the possible level and must have called capabilities and I/O that can ensure that the controller is not an obstacle in the formation of flow. Thus, two of the key performance metrics associated with the SDN controller are set up flow time and the amount of flow per second. This performance metric is very influential when additional SDN controllers must be deployed. For example, if a switch in the SDN architecture starts to provide a flow that is larger than can be supported by an existing SDN controller, so more than one controller must be implemented. The flow setup time and the amount of flow per second are closely related to the latency and throughput generated by the controller. This is also done on the performance test carried out on open daylight (ODL) by SDNTC even though using different tools [3]. The active mode switch consistently initiates traffic to the controller in the form of a packet in message. For most stress tests, MT-collective benchmark (cbench) and multinet switches operate in both active and idle modes [4]. Cbench is a sequential open-source program (parallel application) with multiple datasets (formerly MiDataSets) compiled by the community for realistic benchmarking and research that enables program and architecture optimization. The source code for individual programs has been simplified for portability. Some evolving controllers are ODL [5], pythonic operating system (POX) [6], network operating system (NOX) [7], Beacon [8], Ryu [9], open network operating system (ONOS) [10], floodlight [11], maestro [12], and there are still a few others that continue to emerge and develop. Controller developers develop it by using a variety of platforms, with the use of a variety of platforms this makes each controller has a different performance from each another.

Muntaner tested the NOX, Maestro, Beacon, and Trema controllers. The four controllers are controllers that do not use graphical user interface (GUI) in the performing input table flow, while this study examines controllers that use GUI in the performing input table flow [1]. Research by Turull *et al.* [2] focusing on network virtualization applications and measuring the delay that occurs in internet control message protocol (ICMP) messages, the transfer time of the transmission control protocol (TCP) connections, and packet loss in user datagram protocol (UDP) traffic, various delays between switches and controllers. This research focuses on throughput and latency by simulating the amount of flow per second and simulating the number of switches involved in the floodlight controller and ONOS [13].

Research by Rowshanrad *et al.* [14] tested the performance of floodlight and ODL by measuring the performance of the controller when run on 3 mininet standard topologies namely single, linear, and tree with the number of switches and hosts involved a maximum of 8 (eight) switches. While what is going to be done in this research is to involve as many switches and hosts as possible and measure the flexibility of the controller to run or control a variety of those devices. Therefore, the floodlight and ONOS controllers are tested.

According to Kartadie *et al.* [15] also conducts OF performance testing using OpenWRT-based software. His research shows that the performance of OF switch-based software is often implemented on campus. Testing OpenWRT switching performance supported OF software on campus implementations results in a fluctuating prototype latency value quite diverse compared to mininet. During this study, the performance test used is not a tool test or software-based OpenWRT test but is targeted at a test employing a controller.

Research by Duque *et al.* [16] also tested the performance of floodlight and ODL controllers, Duque's research emphasized the performance of both controllers in handling load balancing by not being attentive to the quantity of devices involved. This study does not focus on load balancing but focuses on the value of the output and latency displayed by both floodlight, ONOS, and ODL controllers with the number of devices involved. Research with almost the same approach was research conducted by Asadollahi and Goswami [17], which performs throughput and latency testing, but only tests one controller, namely floodlight, while researchers conduct tests on a floodlight, ONOS, and ODL controllers and compare the performance of the controller, which one was better.

In line with the research conducted by Zhu [18], throughput and latency are components employed in measuring the performance of the controller. Tools that may be employed in measuring throughput and latency consistent with Zhu include cbench, PktBlaster, and OF network (OFnet) which have their respective advantages. Actual performance testing may be tested using queuing models like those conducted by Xiong *et al.* [19] the research to be allotted adopts this study, but directly compares the two controllers. Other research about controller testing includes testing the performance of floodlight controllers with POX [20] by testing using the topology found within the mininet. This research does not use a particular topology instead puts a burden on the number of devices to the controller. Topology is not the most reference during this study. From some research that has been done, most of them focus on testing the performance or performance of floodlight controllers, and compared to other controllers from different vendors, there are even some that compare with controllers that have different platforms. On the opposite hand, this study also tested the ability of controllers to handle the number of switches involved and also the ability of controllers to handle the flow. The test to be dispensed is to check the throughput and latency by using the common building block (CBB) test tool following the recommendations given by previous researchers. A comparison of features between flood light, ONOS, and ODL obtained from Mamushiane [13] are often seen in Table 1 (adjusted as needed).

Floodlight, ONOS, and ODL are commonly used controllers, so accurate information is needed in this case controller performance, especially if the controller will be used for implementation or other tests, especially on switch usage and flows that may be involved and as a basis for selecting controllers. Testing is limited to using the mininet emulator with OF 1.3 protocol to emulate switches and hosts, and used cbench tool. The controller function in the specification is so important that the performance of the controller is deemed necessary to test how well the controller performs. After testing, it is expected to know the amount of throughput and latency of controllers, especially floodlight, ONOS, and ODL controllers that have the same Java platform.

Table 1. Feature-bade comparison of SDN controllers [13]

	Floodlight	ODL	ONOS
Southbound Interface	OF 1.0, 1.2,1.3,1.4,1.5	OF 1.0, 1.3, NETCONF/YANG, OVSDB,PCEP,BGP/LS, LISP, SNMP, OFCONFIG	OF 1.0, 1.3, 1.4, 1.5,NETCONF
REST API	YES	YES	YES
OS Support	Linux, Windows, MAC	Linux, Windows, MAC	Linux, Windows, MAC
Virtualization	Mininet & OVS	Mininet & OVS	Mininet & OVS
Distribute/Centralize	Distribute	Distribute	Distribute

2. RESEARCH METHOD

In this study, the floodlight, ONOS, and ODL controllers to be tested are run on the host/PC while the Mininet emulator is installed and running on VirtualBox/guest, the OF protocol is used and embedded in mininet is OF 1.3. Floodlight, ONOS, and ODL latency tests with the number of switches ranging from 2 to 450 until the switch limit can no longer be set by the controller. Floodlight, ONOS, and ODL throughput testing with the best number of switches that can still be managed by the controller during latency testing. The test scheme can be seen in Figure 1.

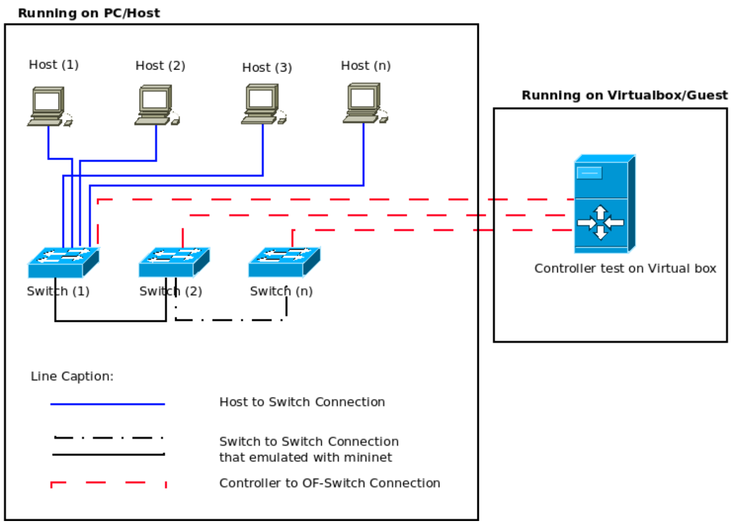
Running on PC/Host
Host (1)
Host (2)
Host (3)
Host (n)
Running on Virtualbox/Guest
Controller test on Virtual box
Switch (1)
Switch (2)
Switch (n)
Line Caption:
Host to Switch Connection
Switch to Switch Connection that emulated with mininet
Controller to OF-Switch Connection

Figure 1. Test scenarios conducted

The specifications of the PC/Laptop used in this test are as follows; i) CPU: intel® CoreTM i3-4005U CPU @1.70 GHz×4 threat, ii) RAM: SODIMM DDR3 4 Gb 1333 MHz, and iii) operation system: Linux UBUNTU 18.04 LTS kernel 4.15.0-47-generic. As shown in Figure 1, the test scenario, testing is completed by using cbench involving a variety of switches and specific hosts (according to the planned research flow diagram) to induce the performance of the controller. The controller was placed on VirtualBox, PC/host run mininet that emulated switches and run cbench. In testing all CPUs threats were used, no changes/variations in CPU, threats were made.

2.1. Latency test

In latency tests, a load with a test length of 5,000 ms is given and a pause start testing after features_reply is received for 5 ms. Latency test is carried out with the following command:

```
˜/oflops/cbench\$ cbench ˜
-c [ip-kontroler] -p 6633 -l 5 ˜
-m 5000 -D 5 -M 5 ˜
-s [number of switches]
```

2.2. Throughput test

This is the same as the latency test, but the throughput test is called with the following command:

```
˜/oflops/cbench\$ cbench ˜
-c [ip-kontroler] -p 6633 -l 5 ˜
-m 5000 -D 5 -M [number of MAC] ˜
-s [number of Switch] -t
```

The test is carried out in the range of 30–450 hosts with the optimal number of switches with an increase in the range of 10 hosts. The two methods above are used in floodlight, ONOS, and ODL tests. Responses/second for latency and throughput-based performance evaluation. this scale is used as a metric to measure these two evaluations. This is because these two evaluations run multiple requests in parallel. Packet_in messages are the only switch-solicited OF packet that should get a response from a controller, so cbench uses a "responses/sec" scale [21].

2.3. Research assumptions used

This study uses several basic assumptions that do not damage or affect the results. This supposal relies on the understanding that topology diagrams are not only within the sort of physical networks and virtual network components but can even include applications that are used, are being developed or applications also can play a job as nodes and links [13]. CPU and memory usage at the time of testing was not tested and is considered not to affect the testing process because it is in a fixed environment.

3. RESULTS AND DISCUSSION

3.1. Latency test

Latency testing is performed on a range of 30 to 450 switches with an increase in the range of 10 switches. The number of tests per session is 5x the number of repetitions. Examples of responses from cbench to testing are as shown in.

```
cbench: controller benchmarking tool
running in mode 'latency'
connecting to
controller at 192.168.56.1:6633
faking 10 switches
offset 1 :: 5 tests
each; 5000 ms per test
with 5 unique source MACs per switch
learning destination
mac addresses before the test
starting test with 5 ms
delay after features\_reply
ignoring first 1 "warmup"
and last 0 "cooldown" loops
connection delay of 0ms
per 1 switch(es)
debugging info is off
01:33:14.386 10 switches: fmods/sec:
6366 3231 3227 3034 3191 5651 3132
3063 2414 2203 total = 7.101693 per ms
01:33:19.488 10 switches: fmods/sec:
7004 3502 3500 3496 3492 6818 3370
3310 2652 2162 total = 7.861158 per ms
01:33:24.590 10 switches: fmods/sec:
7010 3530 3531 3534 3527 6920 3443
3379 2618 2153 total = 7.928978 per ms
01:33:29.694 10 switches: fmods/sec:
7062 3529 3546 3547 3529 6910 3418
3400 2682 2184 total = 7.959643 per ms
01:33:34.797 10 switches: fmods/sec:
```

```
7108 3536 3546 3554 3547 6976 3467
3388 2702 2196 total = 8.002956 per ms
RESULT: 10 switches 4 tests
min/max/avg/stdev =
7861.16/8002.96/7938.18/51.66
responses/s
```

Sample data from the latency test results on the floodlight, ONOS, and ODL controller is seen within the sample data Table 2 where this table provides samples of data obtained in testing with a predetermined switch range. The test value is presented within the sort of a graph that may be seen in Figure 2, the graph will be seen within the response changes every additional number of switches that has got to be handled by controller. In Table 2 when viewed from the average data (avg) responses/second vs switches, the response given by the floodlight controller shows a relatively stable response, even though the ONOS controller gives a high value for a small number of switches, it gives a fluctuating value, until finally, it does not respond to the switch value of 450, while the ODL controller is only able to respond to the switch value of 210, and then does not respond. The average value of latency can be seen in Figure 2, it can be seen from the three controllers, the floodlight controller can provide a more stable latency value compared to other controllers and can continue to respond with an increase in the number of switches and a relatively small decrease in response value compared to ONOS and ODL.

Table 2. Sample data of controllers latency (responses/sec)

		Number of switches				
		30	70	210	360	450
Floodlight controller	min	1789.03	1661.75	1390.09	933.61	185.49
	max	2519.33	2003.42	1729.03	1658.19	1331.69
	avg	2229.46	1831.35	1615.53	1407.83	995.09
		30	70	210	360	450
ONOS controller	min	294,00	1493,97	0,00	0,00	0
	max	12543.25	5465.99	4543.79	4381.88	30
	avg	5859.89	3649.59	1135.95	1468.59	7.5
		30	70	210	360	450
ODL controller	min	1634.4	891.6	5.2	0	0
	max	1986.98	1968.2	76.6	0	0
	avg	1816.24	1634.4	23.85	0	0

A significant decrease in response is given on the number of switches above 210, which can be caused because cbench has an algorithm waiting for a new flow_mod match to respond. As quoted from github mininet, that the latency algorithm in cbench is to simulate n switches (n=16, was the default), then make n OF sessions to the controller. For each CBB session have steps: i) sending a packet; ii) waiting for a suitable flood_mod to return as a response; and iii) repeating steps (1) and (2), then 4) counting the number of times steps (1)-(3) occur per second, so the ability of controllers to respond to data packets only on vulnerable switches 1-50 switches.

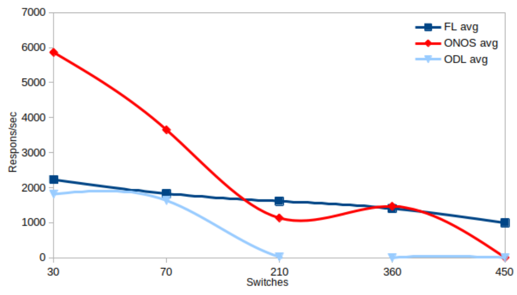

Figure 2. Controller latency average test results (in responses/sec)

3.2. Throughput test

Throughput testing is performed on a range of switches 10 to 450 hosts with an increase in the range of 10 hosts. The number of test switches is 20 switches, each session is repeated five times. Sample data can be seen in Table 3. From the data obtained, a graph was made throughput on controllers as shown in Figure 3.

Table 3. Sample data throughput controllers (responses/sec)

		Number of hosts				
		30	70	210	360	450
Floodlight controller	min	79.06	322.66	419.88	1185.06	1498.26
	max	84.55	326.11	422.04	1190.84	1503.06
	avg	80.62	323.11	421.57	1187.79	1500.38
		30	70	210	360	450
ONOS controller	min	153.85	323.72	249.91	244.12	195.45
	max	246.94	708.97	380.96	323.94	851.59
	avg	200.16	511.41	293.41	267.32	466.09
		30	70	210	360	450
ODL controller	Min	70.4	374.76	817.96	1256.84	1720.32
	Max	95.86	398.66	907.53	1347.97	1798.36
	Avg	85.41	385.62	867.56	1301.11	1747.99

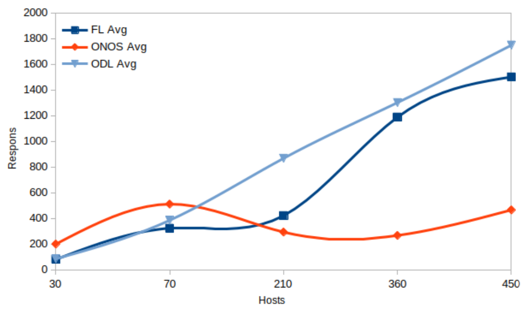

Figure 3. Controller throughput average test results (in responses/sec)

When viewed from the typical data (avg) responses/second vs. the number of hosts shown in Table 3 and Figure 3, the ODL controller can provide good capabilities because with the increasing number of hosts, although floodlight can provide almost the same value, the results provided cannot be said to be stable, the data on the number of hosts 210 floodlights had decreased performance. While ONOS gave relatively lower results than the two controllers, but the given value did not experience a significant increase, so it can be said that this controller experienced a constant response. The upper the quantity of flow/seconds will be served by a controller. The throughput testing algorithm uses cbench [22], [23], declaring in throughput mode (e.g.,, with '-t'): for every session: as long because the buffer isn't full: i) package_in is queued for processing and ii) calculates flow_mod after they return. With a less complicated algorithm, cbench provides a way more stable lead to throughput testing and is directly proportional to the quantity of hosts handled by the controllers.

4. CONCLUSION

The controller can be an important part of the SDN architecture. The controller performance should be tested with various environments. Testing of the controller provides a better view of the choice of controller to be used, although this test still needs improvement, especially in testing with other elements. This test results that the floodlight controller is superior in responding to the latency test compared to the ONOS and ODL controllers, although it provides less stability in the throughput test. The researcher stated that the results of this study were not perfect and could not be used as a reference, but the results of this study could provide a more detailed picture of the controller's performance. The determining element in performance testing is the accuracy in reading the data, and in determining the tool to be used. In selecting a tool, consideration should rest on whether or not it is suitable for the instrument to be tested, and which response controller we are testing. Further research is expected to provide much better results, by replacing better test equipment or by adding test instruments.

REFERENCES

[1] G. R. D. T. Muntaner, "Evaluation of OpenFlow controllers," in *Tech. Rep.*, 2012.
[2] D. Turull, M. Hidell, and P. Sjödin, "Performance evaluation of openflow controllers for network virtualization," *2014 IEEE 15th International Conference on High Performance Switching and Routing (HPSR)*, 2014, pp. 50-56, doi: 10.1109/HPSR.2014.6900881.

[3] SDNCTC, "ONOS controller performance test report," *Global SDN Certified Testing Center*, 2016, pp. 1-18.

[4] I. T. S. Lab, "OpenDaylight performance stress test report," *Tech. Rep.*, 2016.

[5] M. Sisov, "Building a software-defined networking system with OpenDaylight controller," M.S. thesis, Dept. Inf. Technol., Metropolia Univ. of Applied Sciences, Helsinki, Finland, 2016.

[6] S. Kaur, J. Singh, and N. S. Ghumman, "Network programmability using POX controller," in *ICCCS International conference on communication, computing and systems*, 2014, pp. 134-138.

[7] N. Gude *et al.*, "NOX: towards an operating system for networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 3, Jul. 2008, doi: 10.1145/1384609.1384625.

[8] D. Erickson, "The beacon openflow controller," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, 2013, pp. 13-18, doi: 10.1145/2491185.2491189.

[9] "Component-based software defined networking framework: build SDN agilely," *Ryu SDN framework*. [Online]. Available: https://ryu-sdn.org/ (accessed Oct. 08, 2020).

[10] M. Sameer and B. Goswami, "Experimenting with ONOS scalability on software defined network," *Journal of Advanced Research in Dynamical and Control Systems*, vol. 10, no. 14, Jan. 2018, pp. 1820–1830.

[11] "Floodlight controller," *Project Floodlight*, 2016. [Online]. Available: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview (accessed Oct. 08, 2020).

[12] Z. Cai, A. L. Cox, and T. S. E. Ng, "Maestro: a system for scalable OpenFlow control" *Rice University*, 2010, pp. 1-10.

[13] L. Mamushiane, A. Lysko, and S. Dlamini, "A comparative evaluation of the performance of popular SDN controllers," *2018 Wireless Days (WD)*, 2018, pp. 54-59, doi: 10.1109/WD.2018.8361694.

[14] S. Rowshanrad, V. Abdi, and M. Keshtgari, "Performance evaluation of SDN controllers: floodlight and OpenDaylight," *IIUM Engineering Journal*, vol. 17, no. 2, pp. 47–57, 2016, doi: 10.31436/iiumej.v17i2.615.

[15] R. Kartadie, F. Rozi, and E. Utami, "Openflow switch software-based performance test on its implementation on campus network," *Journal of Theoretical and Applied Information Technology*, vol. 96, pp. 4136–4146, Jul. 2018.

[16] J. P. Duque, D. D. Beltr´an, and G. P. Leguizam, "OpenDaylight vs. Floodlight: Comparative Analysis of a Load Balancing Algorithm for Software Defined Networking," *International Journal of Communication Networks and Information Security (IJCNIS)*, vol. 10, no. 2, Apr. 2022. doi: 10.17762/ijcnis.v10i2.3178.

[17] S. Asadollahi and B. Goswami, "Experimenting with scalability of floodlight controller in software defined networks," *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, 2017, pp. 288-292, doi: 10.1109/ICEECCOT.2017.8284684.

[18] L. Zhu, M. M. Karim, K. Sharif, F. Li, X. Du, and M. Guizani, "SDN controllers: benchmarking & performance evaluation," in *arXiv-Networking and Internet Architecture*, pp. 1–14, 2019, doi: 10.48550/arXiv.1902.04491.

[19] B. Xiong, K. Yang, J. Zhao, W. Li, and K. Li, "Performance evaluation of OpenFlow-based software-defined networks based on queueing model," *Computer Networks*, vol. 102, pp. 172–185, Jun. 2016, doi: 10.1016/j.comnet.2016.03.005.

[20] I. Z. Bholebawa and U. D. Dalal, "Performance analysis of SDN/OpenFlow controllers: POX versus floodlight," *Wireless Personal Communications*, vol. 98, no. 2, pp. 1679–1699, 2018, doi: 10.1007/s11277-017-4939-z.

[21] R. Sherwood, "oflops/cbench at master-mininet/oflops," *GitHub*, 2013. [Online]. Available: https://github.com/mininet/oflops/tree/master/cbench (accessed Oct. 16, 2019).

[22] "libfluid: benchmarks," *GitHub*. [Online]. Available: https://opennetworkingfoundation.github.io/libfluid/md_doc_Benchmarks.html (accessed Sept. 01, 2021).

[23] R. Jawaharan, P. M. Mohan, T. Das, and M. Gurusamy, "Empirical evaluation of sdn controllers using mininet/wireshark and comparison with cbench," *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, 2018, pp. 1-2, doi: 10.1109/ICCCN.2018.8487382.

BIOGRAPHIES OF AUTHORS

Rikie Kartadie master of Computer Science from Universiti Amikom Yogyakarta (2014). Now a lecturer at the Department of Computer Engineering, Universitas Technologi Digital Indonesia. Research interests are computer networks, SDN, and IoT, teaching in the field of networking. Have received several grants from the Ministry of Research, Technology and Higher Education. Additional duties as head of information systems and publications of LPPM-UTDI. He can be contacted at email: rikie@utdi.ac.id.

Edy Prayitno master of Engineering from Universitas Gadjah Mada Yogyakarta (2010). Now a lecturer at the Departement of Information System, Universitas Teknologi Digital Indonesia. Research interests are information system, IoT, and computer network. He received several grants from the Ministry of Education and Culture of the Republic of Indonesia. He can be contacted at email: edyprayitno@utdi.ac.id.